WIND TURBINE SCADA ALARM PATTERN RECOGNITION

B. Chen*, Y.N. Qiu, Y. Feng, P.J. Tavner, W.W. Song

**School of Engineering and Computing Science, Durham University, UK, bindi.chen@dur.ac.uk*

Keywords: SCADA Alarms, Wind Turbine, Reliability, Artificial Neural Network, Back-propagation network

Abstract

Current wind turbine (WT) studies focus on improving their reliability and reducing the cost of energy, particularly when they are operated offshore. WT Supervisory Control and Data Acquisition (SCADA) systems contain alarm signals providing significant important information. Pattern recognition embodies a set of promising techniques for intelligently processing WT SCADA alarms. This paper presents the feasibility study of SCADA alarm processing and diagnosis method using an artificial neural network (ANN). The back-propagation network (BPN) algorithm was used to supervise a three layers network to identify a WT pitch system fault, known to be of high importance, from pitch system alarm. The trained ANN was then applied on another 4 WTs to find similar pitch system faults. Based on this study, we have found the general mapping capability of the ANN help to identify those most likely WT faults from SCADA alarm signals, but a wide range of representative alarm patterns are necessary for supervisory training.

1 Introduction

Current studies of wind turbines focus on improving reliability [4,9,10] because good wind turbine reliability, together with predictable turbine maintenance schedule, will result in a reduced cost of energy, which will determine the success of a wind farm project. This is even more important for offshore wind farms due to their high initial capital cost and limited accessibility, causing higher operational and maintenance (O&M) cost [11,14] and prolonged capital payback.

The essence of improving the reliability of wind turbine is to reduce the downtime and increase its availability by optimizing both the wind turbine design and the maintenance schedule. Both of these strategies require a full understanding of the wind turbine system and a detailed analysis of its failure mechanisms. WT SCADA systems provide a rich resource to achieve this capability as it archives comprehensive signal information, historical alarms and detailed fault logs, as well as environmental and operational conditions [1,2,6,8]. A wind turbine's systematic performance can be monitored through a proper analysis of the information collected by the SCADA system which covers all the major WT sub-assemblies. Initial attempts to use SCADA alarms to detect wind turbine failure are made recently [15,17]. This paper focuses on using pattern recognition to analyse SCADA alarm and proves its feasibility on wind turbine fault diagnosis, concentrating particularly on a WT electric pitch system, which are known to be fault prone and which produce a significant number of SCADA alarms [17].

2 SCADA Alarms

For a wind turbine, many individual systems are installed to monitor its running situation which includes SCADA and Condition Monitoring (CMS) systems. CMS is designed to detect the incipient failure of WT components [5] with higher frequency bandwidth than SCADA, > 10 kHz, and higher costs per channel and it usually monitors the main drive chain area. The SCADA signal bandwidth is low, 10 minutes samples, and data storage requirement is less than CMS. Alarm information is stored and annunciated by the SCADA system. Typically, alarms are used to indicate the need for operator's emergency action to protect a WT from running into a risky condition. Whenever the WT malfunctions, associated alarms will be triggered.

A recent study from [17] has investigated the Key Performance Indicators (KPI) of alarms from 4 onshore, 30-40 WTs wind farm. The results show that an average alarm rate varying from 4-20 per 10 minutes and maximum alarm rate varying from 390-1,500 per 10 minutes. These are very high figures from relatively small onshore wind farms and the alarm rate would need to be reduced to be interpretable by operators or maintainers. In 2011 [15] introduced time-sequence and probability-based analysis method to analyse SCADA alarm data. These two methods have proved to be potential for rationalising and reducing alarm data providing fault detection, diagnosis and prognosis from the conditions generating the alarms.

This paper proposes a novel method which is based on artificial neural network to realize WT failure diagnostic. From system engineering point of view, the alarm generation due to a fault can be described as:

$$C_{fault} \rightarrow P_{alarm} \quad (1)$$

which means fault C generates the alarm pattern P. The task of diagnosing alarms can be considered as the inversion of the alarm generation task:

$$P_{alarm} \rightarrow C_{fault} \quad (2)$$

where P is the incoming alarm pattern, which may be unknown and C is the system fault most likely to have occurred. Therefore, alarm diagnosis using ANN is intended to identify the most likely alarm pattern from incoming alarms patterns and use it for diagnosis.

3 Artificial Neural Networks

An ANN is a computational model made up of many processing neurons that has a natural propensity for storing experiential knowledge and making it available for use [12]. Typically, the structure of an ANN consists of three layers, as shown in Figure 1. The first layer consists of input nodes, which are connected to the neurons of a hidden layer. The hidden neurons are connected to the neurons of the third or output layer.

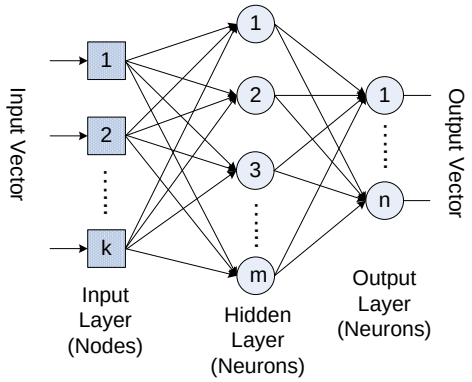

Figure 1: A $k \times m \times n$ feed-forward ANN.

The knowledge of training data is stored in the connection strengths, known as weights, between the processing elements from layer to layer during the training process [12]. The network is trained in accordance with a training algorithm, which governs how connection weights are modified and adjusted in response to the training data feed into the input nodes and the desired outputs at the output neurons. In the recall process, the trained ANN accepts inputs presented at the input nodes and then produces a response at the output layer [13].

Generally, the relation of a three layer $(k \times m \times n)$ feed-forward ANN can be represented in vector notations [15]. If I is a k dimensional column vector presenting inputs and H is a m dimensional column vector representing the results of hidden layer, then:

$$H = f_1(W_1 I) \qquad (3)$$

where W_1 is a $m \times k$ weight matrix assigned to the connection between the input and hidden layers, and f_1 is an activation function using a sigmoid function, as follows:

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad (4)$$

This function has the ability to produce continuous non-linear threshold function and transforms inputs between $-\infty$ and $+\infty$ into real numbers between 0 and 1 [7]. Similarly, the n dimensional column vector for the output layer can be represented as follows:

$$O = f_2(W_2 H) = f_2\big(W_2 f_1(W_1 I)\big) \qquad (5)$$

where f_2 is another activation function using a sigmoid function as shown in (4), and $W_2$is a $n \times m$ weight matrix for the connection between the hidden and output layer. As we can see from (5), the calculations of the output of the ANN involve two weight matrix multiplications and two applications of the activation function. Therefore, the use of this ANN will require some computational overhead.

3.1 Back-propagation network Training Algorithm

A back-propagation network (BPN) training algorithm is a common method for teaching feed-forward ANNs how to perform a given task [16]. This algorithm adopts a training rule called the Generalized Delta Rule (GDR), which follows an iterative gradient descent algorithm designed to minimize the overall mean square error E, defined as:

$$E = \frac{1}{2N}\sum_{n=1}^{N}\|D_n - O_n\| \qquad (6)$$

where N denote the number of training pattern presented to the input layer. D_n represents the desired output of the nth input pattern and O_n is the actual output of the same input pattern (Note: Both of D_n and O_n are vectors). The update of ANN weight is calculated by using the following equation:

$$w_{n+1} = \eta\delta x + \alpha w_n \qquad (7)$$

where n denotes the presentation step, η is the training rate and α is the momentum coefficient. By introducing the training rate and momentum coefficient, training speed can be increased without oscillation [13]. The error signal δ at the jth neuron is determined as follows:

$$\delta_j = \begin{cases} x_j(1-x_j)\sum_k \delta_k^{(L+1)} w_{k,j}^{(L+1)}, & in\ hidden\ layer \\ x_j(1-x_j)(d_j - x_j), & in\ ouput\ layer \end{cases} \qquad (8)$$

where d_j is the desired output of the jth neuron, x_j is the output of the connected neuron, $L+1$ means the next layer – the output layer.

Using this algorithm, the ANN is trained by initialising small random weights and then presenting all training patterns repeatedly. Weights are modified and adjusted after every presentation step, by propagating error signals from neurons in the output layer to neurons in the hidden layer and nodes in the input layer. Finally, if the error E in (6) is less than a specific minimum value, training can be terminated and the ANN is ready for use [14, 16].

4 Construction of ANN

4.1 Get training data – Pitch System fault

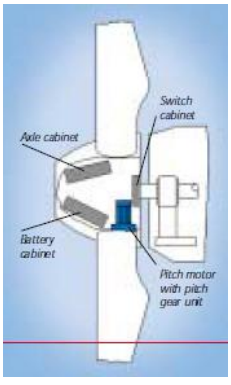

Figure 2: Wind turbine pitch motor (In blue)

As shown in Figure 2, the modern 3-bladed WT has a pitch motor in each blade to control the blade angle, extract optimum power from the wind and avoid rotor overspeed. As the rotor speed increases, the blade pitch angle increases to control WT toque. In a WT without any pitch faults, three pitch motor torques should be identical. Any detectable large torque difference between blade pitch motors can be caused by a possible pitch system fault. In order to get training data for the ANN, 3 criteria were defined to identify pitch system faults and then use them to obtain corresponding alarm patterns. The 3 cirteria are listed below and an example is shown in Figure 3 with the relevant SCADA signal descriptions listed in Table 1.

- Identification of a monitoring period;
- Significant pitch motor torque difference during this period;
- No significant change in wind speed during this period.
- It has maintenance records during this period.

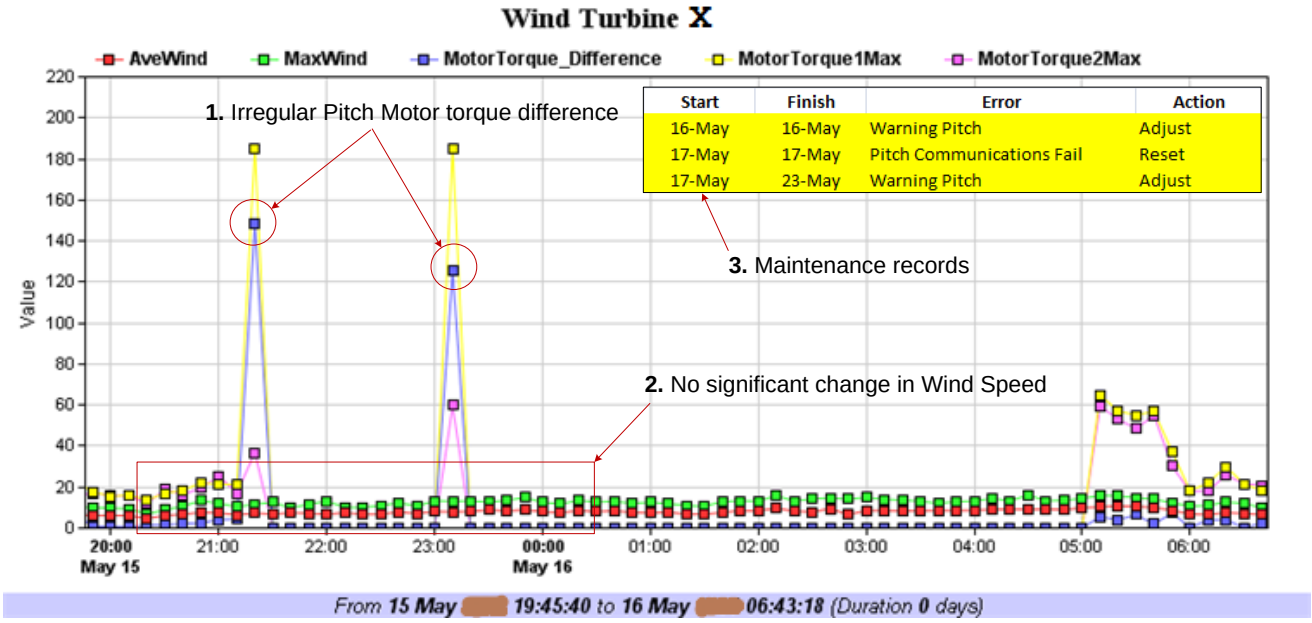

Figure 3: Criteria to identify a pitch system fault (Vertical axis is the value of different SCADA signals; Horizontal axis is the date time)

SCADA Signals	*Description (Over the past 10 minutes)*
AveWind	The average wind speed
MaxWind	The maximum wind speed
MotorTorque1Max	The maximum pitch motor torque in blade 1
MotorTorque2Max	The maximum pitch motor torque in blade 2
MotorTorque_Difference	Pitch motor torque difference between blade 1 and blade 2

Table 1: The corresponding SCADA signals description in Figure 3.

Subject to above requirements, the corresponding alarm pattern were found and shown in Figure 4. The alarm fault behaviours can be transformed into an alarm matrix, which is obtained by putting the data for each fault into an alarm input vector as follows:

$$P = [A_1, A_2, A_3, \dots, A_{31}]^T$$

where P can be considered to characterize pitch system fault and A takes one of the values 0 and 1; both indicate the on and off state of the relevant alarm. These alarm patterns represent pitch system faults whereas the reminders represent no pitch system fault.

By analysing a WT X with pitch problem over a period of 26 months, we found:

- 31 alarms are associated with this pitch system fault;
- 5,760 alarm triggering frequency associated with this pitch system fault;
- 221 different alarm patterns;
- Among these 221 alarm patterns:
  - 15 alarm patterns are associated with this pitch system fault.
  - 206 alarm patterns are not associated with this pitch system fault.

Then, using:

- "1 0" to represent the desired output of this pitch system fault;
- "0 1" to stand for the desired output of no this pitch system fault;

The 221 input/output pairs of training data were transformed and listed in Table 2.

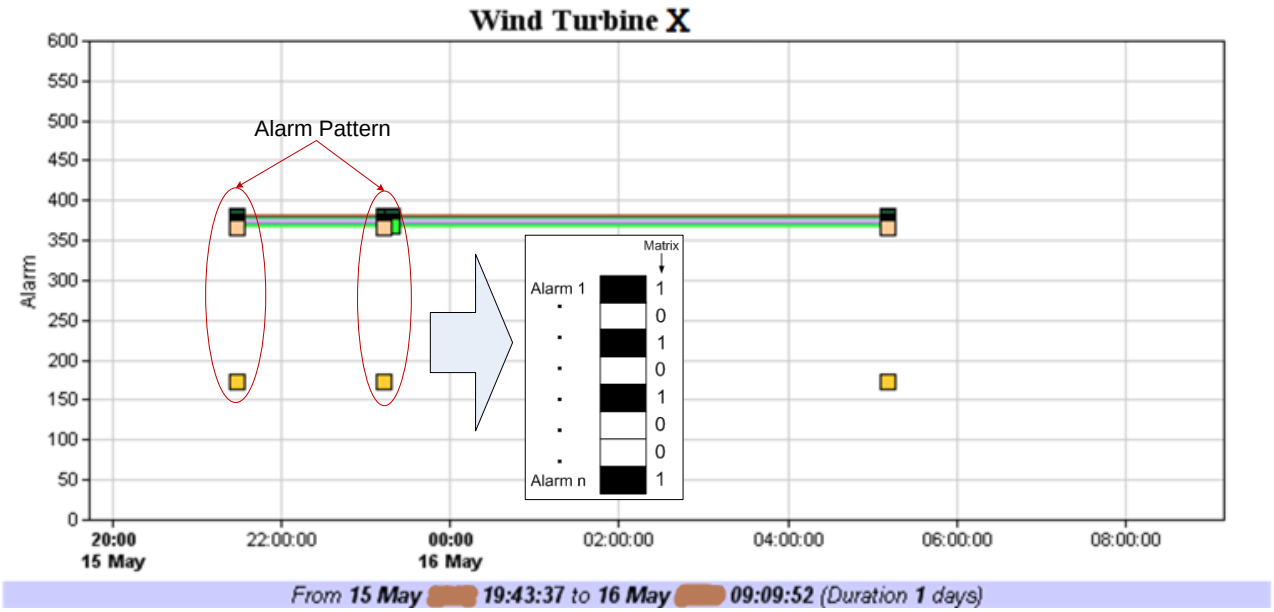

Figure 4: Corresponding alarm signals from WT X from Figure 3.

	Alarm Pattern (Inputs – 31 alarms)	*Desired Output*	*Description*
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 1 0 0 0 1 0 1	1 0	Pitch System fault
2	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 1 0 1 1 1 0	1 0	
	……	1 0	
15	1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 1 0 0	1 0	
16 - 221	Other 206 alarm patterns	0 1	No Pitch System fault

Table 2: Training data

4.2 ANN Model

In the ANN implementation, an input layer consist of 31 alarm inputs, a hidden layer consists of *j* neurons and an output layer consists of 2 fault identification neurons, as shown in Figure 5 is constructed. In addition, in order to control the behaviour of the ANN layers and improve the accuracy of the ANN output, a special processing element called bias with a constant input +1 is attached to all of the neurons in hidden and output layers. This network model represents the mapping from alarm pattern space to a fault space.

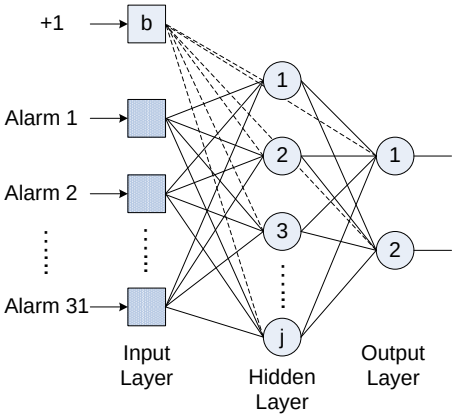

Figure 5: Implemented ANN model

4.3 Training the ANN

The training process of the ANN is shown in Figure 6. During the training, an alarm input vector was fed into the input layer of the ANN and the desired output corresponding to input vector was used to compare with the actual ANN output. If the result of the comparison was unacceptable, the BPN training algorithm adjusted the weights to be consistent with the imposed input vector and desired output. The weights were then readjusted to accommodate new input vector with the corresponding desired output. The training process is repeated until the overall mean square error E in (6) was less than a specific minimum value. In this work, the minimum value was set to 0.0001 to achieve adequate network performance [13].

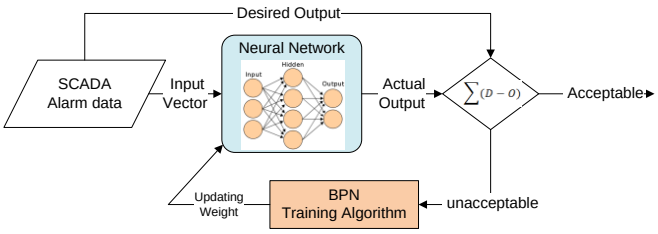

Figure 6: Training process

The training was performed on a Server with 2 processors, 48G memory and 8TB hard drive and using C# programming language. The weight and biases were randomly initialised from -1 to 1 at the beginning, and then updated after every training cycle using (7). Based on [7], the training rate η and the momentum coefficient α were set to 0.7 and 0.9, respectively.

5 Results and Discussion

By choosing different number of neuron j in the hidden layer, as shown in Figure 5, 3 different ANN models were constructed. They were $31 \times 30 \times 2$, $31 \times 50 \times 2$ and $31 \times 70 \times 2$. The time to achieve the mean square error to 0.0001 is shown in Table 3. In this work, the optimum number of hidden layer neurons was 50.

NN Model	*No. of training cycles*
$31 \times 30 \times 2$	110
$31 \times 50 \times 2$	105
$31 \times 70 \times 2$	142

Table 3: Training cycles to achieve mean square error to 0.0001.

After the network has been fully trained, fault diagnosis was simply a matter of presenting a new alarm input vector data to the input layer of the trained ANN and calculating the ANN output. In other words, a fault can be identified when the ANN output is close to "1 0", for example "0.9983 0".

In order to demonstrate the feasibility of the proposed ANN, the trained ANN was applied in a test to another 4 WTs to identify similar pitch system faults found in Table 2. The intention was to determine:

a) Can a defective pitch mechanism be detected by the ANN;
b) How many hidden layers give the best result;

The data about pitch faults was contained in a Maintenance Log and the above test used the results from that Maintenance Log to determine whether the trained ANN was giving effective results on the 4 WTs.

The results are summarised in Table 4 where a was the number of pitch events in the Maintenance Log, b was the number of incidents events identified by the ANN, d was the number of pitch events correctly predicted by ANN, checked by pitch significance and date in the Maintenance Log.

		Number of Hidden Neurons (j)								
		30			50			70		
WTs	*a*	*b*	*d*	*d/b*	*b*	*d*	*d/b*	*b*	*d*	*d/b*
X (trained)	102									
A	53	33	15	45%	32	15	47%	31	14	45%
B	67	30	10	33%	25	9	36%	23	7	30%
C	20	23	3	13%	23	3	13%	22	2	9%
D	33	24	2	8%	24	2	8%	23	2	9%

Table 4: Accuracy rate of the different trained ANNs (j is the No. of the neurons in hidden layer).

The Table shows first of all that 50 hidden layers is better than 30 but that 70 hidden layers appears to give no additional benefit. Therefore, by considering both Tables 3 & 4, we believe 50 neurons in hidden layer is the best ANN solution to detect WT pitch system fault using 31 pitch system alarms.

The results also show that the ANN correctly identifies WTs A & B as experiencing more pitch events than WTs C & D. This may be due to WTs A & B having similar running environments to the training used WT X.

In this work, the trained alarm patterns, previously listed in Table 2 were constructed using pitch system faults occurred in a real WT. These alarm patterns represent actual alarm patterns from one WT, although the number of possible alarm patterns for the 31 pitch system alarms is theoretically up to 2^{31}. In fact, other alarm patterns may be possible due to the sensor failures, multiple faults or dynamic system behaviour. Although these untrained alarm patterns do occur, the ANN's general mapping capability enables to identify the most likely faults. However, if a new alarm pattern is widely different from the trained alarm patterns, serious problems may occur

due to excessive extrapolation errors [7]. For better performance, large numbers of training data covering a wider range of representative alarm patterns are needed to assure adequate network performance.

A drawback of using ANN we found in this work is that the result of ANN is highly dependent on the precision of the training data. In addition, different WT manufacturers might have different type of alarm data and this will increase the difficulty of promoting ANN on WT fault diagnosis.

6 Conclusion

This paper presented a feasibility study of WT alarm processing and diagnosis using ANN. The suitability study of the ANN for the diagnosis of WT pitch system faults from pitch alarm has been demonstrated. The results show that:

- ANN is a feasible method for online WT fault diagnosis.
- ANN has potential to fast identify WT failures and reduce the alarm rate.
- The confidence of ANN can be further improved by adding more well-defined training patterns.

In conclusion, the NN approach has the potential to rationalise and reduce alarm data and provide valuable fault detection, diagnosis and prognosis. The ANN based systems can run very fast if hardware implementations are becoming available. This would make the system especially suitable for real-time WT fault diagnosis.

Acknowledgements

The authors acknowledge the EU FP7 Project RELIAWIND 212966 and UK EPSRC SuperGen Wind EP/H018662/1 Projects. The data used in the paper derived from SCADA data from a number of different wind farm sources.

References

[1] A. Zaher, S.D.J. McArthur, D.G. Infieeld, Y. Patel (2009). Online Wind Turbine Fault Detection Through Automated SCADA Data Analysis. *Wind Energy*, **12**(6) , 574-593.

[2] C. S. Gray and S. J. Watson (2009). Physics of Failure Approach to Wind Turbine Condition Based Maintenance. *Wind Energy*.

[3] D. Hammerstrom, (1993). Working with neural networks. *IEEE Spectrum*, pp. 46–53 July

[4] F. Spinato, P.J. Tavner, G.J.W. van Bussel, E. Koutoulakos (2009). Reliability of wind turbine Subassemblies. *IET Renewable Power Generation*, **3**(4) , 1-15.

[5] Germanischer Lloyd, *Rules and Guidelines, IV Industrial Services, 4 Guideline for the Certification of Condition Monitoring Systems for Wind Turbines*, 2007

[6] H Stiesdal, P.H.-M., (2005). Design For reliability, in Copenhagen Offshore Wind 2005. 2005: Copenhagen.

[7] J. Heaton (2008) *Introduction to Neural Networks for C#*, 2nd ed. Publisher: Heaton Research, Inc.

[8] K. Andrew, V. Anoop, (2010). The future of wind turbine diagnostics. *Wind Systems Magazine*. April 2010.

[9] M. R. Wilkinson, et al. (2010). Methodology and results of the ReliaWind reliability field study *In: EWEC, 2010*, April 21-24, Warsaw Poland.

[10] P.J. Tavner, J.P. Xiang, F. Spinato, (2006). Reliability analysis for wind turbines, *Wind Energy*, **10**(1), July 2006.

[11] S. Faulstich, B. Hahn and P. J. Tavner, (2009). Wind turbine downtime and its importance for offshore deployment, *Wind Energy*, Wiley Interscience.

[12] S. Haykin (2009). *Neural Networks and Learning Machines*, 3rd ed., Publisher: Prentice Hall.

[13] S.W. Cheon, S.H. Chang, H.Y. Chung, Z.N. Bien (1993). Application of Neural Networks to Multiple Alarm Processing and Diagnosis in Nuclear Power Plants. *IEEE Transactions on Nuclear Science*, **40**(1), 1993

[14] Y Feng, P.J. Tavner, H. Long, (2010). Early experiences with UK round 1 offshore wind farms, *Institution of Civil Engineering, Energy 163*, Issue EN4, pp. 167–181

[15] Y.N. Qiu, P. Richardson, Y. Feng, P.J. Tavner, G. Erdos, (2011). SCADA Alarm analysis for improving wind turbine reliability. *EWEA 2011*, Brussels.

[16] Y. Chauvin, D.E. Rumelhart (1995). *Backpropagation: Theory, Architecture, and Application.* Lawrence Erlbaum Associates, Inc., Publishers.

[17] Y.N. Qiu, Y. Feng, P.J. Tavner, P. Richardson, G. Erdos, B. Chen, (2011). Wind Turbine SCADA Alarm Analysis for Improving Reliability. Accepted for publication in *Wind Energy*.